IEEE International Conference on Data Engineering

On the Efficiency of Provenance Queries

Anastasios Kementsietsidis
IBM T.J. Watson Research Center
19 Skyline Dr., Hawthorne, NY, USA
akement@us.ibm.com

Min Wang
IBM T.J. Watson Research Center
19 Skyline Dr., Hawthorne, NY, USA
min@us.ibm.com

Abstract— **While models for data provenance have been extensively studied in the literature, the *efficient* evaluation of the resulting provenance queries remains an *open problem*. Traditional query optimization techniques, like the use of general-purpose indexes, or the materialization of provenance data, fail on different fronts to address the problem. Provenance-specific optimization techniques, like the use of customized indexes, similarly prove inadequate since the techniques are bound to specific provenance models. Therefore, the need to develop *generic provenance-aware* techniques quickly becomes apparent.**

In this paper, we argue for such a generic technique in the form of a provenance index structure that can be used to efficiently evaluate provenance queries in a variety of contexts. By highlighting the limitations of existing techniques, we identify the set of key properties of the generic index, including a *novel* property called *duality* which guarantees that the *single* index can evaluate both *backward* provenance queries (which data items from a set I are associated with an item from set O) and *forward* provenance queries (which items from O are associated with an item from I).

I. Introduction

Provenance [1] (*a.k.a.* lineage [2], or pedigree) attempts to capture the *processing history* of a data item along with the *inter-dependencies* that arise during the processing between different data items. As a research topic, provenance is currently receiving considerable attention in the literature (see [3][4] for surveys of recent works). To a certain extent, this attention is motivated by a trend in many emerging real-life applications (e.g., in domains like healthcare [5], banking/finance, the life sciences [6]) which dictates that the data processing and analytics needs of these applications require (or even demand) provenance information. As an example of such an application, at the IBM Century [5] project, we are building a framework for healthcare online analytics of sensor-based medical data, in which provenance plays a central role [7]. In more detail, a typical scenario in Century considers a patient, say *"John Doe"*, whose medical data (e.g., blood-pressure, heart-rate, SpO2, electrocardiogram readings) are streamed into and analyzed by the Century analytics in *real-time* (see Figure 1). Potential outputs of these analytics might include (among other things) a medical alert for John, which is forwarded to John's doctor and might include a medical recommendation (e.g., reduction of medication dosage). Since it is *imperative* for the doctor to accurately evaluate the validity and severity of this alert, Century is *required* to provide a pair of (complementary) query services to compute the alert's *workflow* [3] and *data provenance* [3]. Specifically, for a given output (e.g. an alert), the workflow provenance query service returns the analytics workflow that results in this output. So,

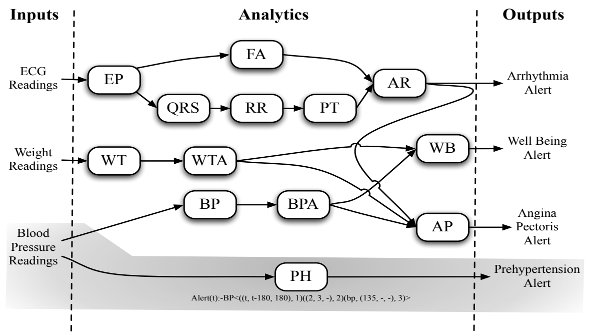

Fig. 1. Sample analytics graph

for example, given an "Angina Pectoris" alert, the workflow provenance query should return the analytics sub-graph of Figure 1 that results in the alert, i.e., the whole analytics graph minus the "WB" and "PH" nodes. The data provenance query service returns the specific input (and intermediate result) medical data that were analyzed in this workflow and contributed to the output generation. So, for the "Angina Pectoris" alert, the data provenance query should return not only the input ECG, weight and blood pressure readings, but also the intermediate data that result from the analysis of these readings (e.g., the outputs of analytics nodes like the "EP", "QRS", "WT", "BP", etc.). Clearly, workflow and data provenance are not specific to Century or healthcare. Workflow provenance has also been investigated in the context of scientific workflows [8][9], while a large body of works has considered various types [10] and models [11][12][13] to capture data provenance.

Irrespectively of the application domains and of the types and models of provenance considered in each one, provenance-related information (both for workflow and data provenance) is stored in a database alongside the base application data. Therefore, to determine the provenance of a particular data item ultimately one has to evaluate one, or more, *provenance queries*. Since provenance usually captures intricate inter-relationships between data items, the corresponding provenance queries that need to be evaluated are often quite complex [11][13]. Traditional optimization techniques that rely on the optimizer, or on generic indexes, usually prove inadequate in this context, and often the development of customized optimization techniques proves necessary [11][13][14]. Yet, these techniques are not generic enough to be used across different provenance models and as a result the efficient evaluation of provenance queries remains an *open problem* [15].

To this end, in this paper we argue for a *generic* technique in the form of a provenance index structure that can be used to efficiently evaluate provenance queries in a variety of contexts. We consider an index structure which is particularly suited for data provenance queries but, as it becomes clear later on, it can also be used for workflow provenance queries. Our motivation for focusing on the former type of provenance queries is that data provenance is characterized as *fine-grained* [3], when compared to workflow provenance, and the data required/accessed while answering data provenance queries are expected to be orders of magnitude more than those for workflow provenance queries. Therefore, efficiency is more imperative for the former queries.

II. Motivating Example

In the following paragraphs, we use an example from the Century system to illustrate some of the challenges in evaluating (data) provenance queries and some of the limitations of the standard evaluation and optimization techniques. In Figures 2(a) and (b) we show sample relations used in Century to store blood-pressure (BP) readings and *"Prehypertension"* alerts. Using as input the blood-pressure readings, Century analytics (see shaded area in Figure 1) can *generate* a *"Prehypertension"* alert using the following procedure: With 4 readings in every 3-hour epoch (readings and alerts in the same epoch are colored alike), an alert is generated at the end of an epoch when any of the readings in the middle of an epoch (the 2nd and 3rd reading) has a systolic pressure larger than 135mmHg. Intuitively, assuming some medication is administered to a patient at the beginning of each 3-hour epoch, these analytics check whether the medication affects a patient's blood pressure (assuming any such effects should be visible in the middle of the epoch, where medication might be more *active*). Given these analytics, how can we model the provenance of a generated alert, i.e., its inter-dependencies with the input readings? In Century, the TVC [16] provenance model is used to establish the inter-relastionships between the inputs and outputs of the Century analytics and we briefly describe the model next.

A. The TVC provenance model

The TVC model [16] is used to represent provenance associations in a wide spectrum of real-life applications. Although a number of similar models can be found in the literature, the TVC model is just used for convenience here. As we show in the following paragraphs, our conclusions are not specific to the TVC model and similar conclusions can be drawn by using any other provenance model.

In a number of applications, special analytics process input data from a set $\mathcal{I}$ (e.g., medical readings, stock quotes) and produce a new set of output data $\mathcal{O}$ (e.g., medical alerts, stock buying/selling recommendations). Although the precise nature of these analytics is application dependent, a key observation is that the provenance relationship between the inputs and outputs of the analytics can be described in terms of some invariant primitives [16]. In what follows we describe three such basic primitives:

Time (T): This primitive is used when a data item from $\mathcal{O}$ is associated with data items from $\mathcal{I}$ that are bounded by a time window. This is the case in our motivating example where a medical alert is associated with the blood-pressure readings of the last epoch. The primitive format is $\mathcal{O}(t)$:- $\mathcal{I}\langle((t-t_b, t-t_e, sf), or)\rangle$, where t specifies the timestamp of the data item of $\mathcal{O}$, $(t-t_b)$ and $(t-t_e)$ the time window enclosing the items from $\mathcal{I}$, sf the *shift* of the time window between consecutive items in $\mathcal{O}$, and or the primitive order, when multiple primitives for the *same* item in $\mathcal{O}$ are given (more on this later). To illustrate, rule $Alert(t)$:- $BP\langle((t, t-180min, 180min), 1)\rangle$ indicates that an alert at time t is associated with all the blood pressure readings in a 3-hour epoch, with consecutive epochs having no time overlap.

Sequence (S): The primitive expresses dependencies in terms of a data item in $\mathcal{O}$ and sequences of data items in $\mathcal{I}$. Its format is $\mathcal{O}(t)$:- $\mathcal{I}\langle((s_b, s_e, sf), or)\rangle$, where s_e and s_e specify the start and end sequence numbers of data items in $\mathcal{I}$, sf the *shift* of the sequence window between consecutive items in $\mathcal{O}$, and or the primitive order. For example, rule $AVGBP(t)$:- $BP\langle((1, 10, 5), 1)\rangle$ indicates that a blood-pressure reading average at time t is calculated by the last 10 blood-pressure readings (irrespectively of their timestamps). Here, the sequence window is shifted by 5 readings and therefore, the same reading in $\mathcal{I}$ might contribute to the generation of two different averages in $\mathcal{O}$.

Value (V): The primitive expresses dependencies in terms of a data item in $\mathcal{O}$ and a set of items in $\mathcal{I}$ whose attribute *attr* has a value in a predefined range. The format is $\mathcal{O}(t)$:- $\mathcal{I}\langle(attr, (v_b, v_e, sf), or)\rangle$, where v_b and v_e specify the range of values the attribute $attr$ must satisfy, sf the *shift* of the value window, and or the primitive order. For example, rule $Alert(t)$:- $BP\langle(systolic, (135, 140, 10), 1)\rangle$ indicates that an alert at time t depends on all the blood-pressure readings between 135 and 140mmHg. Between consecutive alerts the *ten* oldest readings are dropped from consideration.

For significantly enhanced expressiveness, a combination of primitives can be used and the unique 'order' field defines an evaluation order of the primitives, with the output of a lower order primitive acting as the input for a higher order primitive. For example, the rule:

$$Alert(t)\text{:-}BP\langle((t, t-180, 180), 1)((2, 3, -), 2)(bp, (135, -, -), 3)\rangle$$

expresses the complex association of alerts and blood pressure readings from our motivating example where a *"Prehypertension"* alert is generated from 4 readings in every 3-hour epoch, if the 2nd or 3rd reading in the epoch have a systolic pressure larger than 135mmHg. In Century, a TVC formula like the one shown above is associated with each node in the analytics workflow (as shown in Figure 1 for node "PH" – due to lack of space, the formulas for the remaining nodes are omitted).

B. Provenance Queries

In Century, given a *"Prehypertension"* alert for patient John, his doctor must be able to issue a provenance query to

bpID	*tm*	*systolic*	*diastolic*
101	13:10	130	79
102	14:00	136	81
103	14:36	137	82
104	15:40	138	82
105	16:12	137	81
106	17:23	134	81
107	18:07	134	82
108	18:46	131	79
109	19:16	131	80
110	19:58	136	82
111	20:49	134	81
112	21:43	135	79
113	22:26	135	80
114	22:51	137	82
115	23:17	136	81
116	23:49	135	79

(a) The *BP* relation

alID	*tm*	*alert*
201	16:00	"Prehypertension"
202	22:00	"Prehypertension"
203	00:00	"Prehypertension"

(b) The *Alert* relation

bpID	*bptm*	*alID*	*altm*
102	14:00	201	16:00
103	14:36	201	16:00
111	20:49	202	22:00
114	22:51	203	00:00
115	23:17	203	00:00

(c) The *BP2Alert* relation

Fig. 2. Blood pressure readings and alerts

```
Q_B^M: SELECT * FROM BP WHERE BP.bpID IN
         (SELECT bpID FROM BP2Alert WHERE BP2Alert.altm = ⟨al_tm⟩)

Q_B^M: SELECT * FROM Alert WHERE Alert.alID IN
         (SELECT alID FROM BP2Alert WHERE BP2Alert.bptm = ⟨bp_tm⟩)

Q_B^E: SELECT * FROM
         (SELECT * FROM
           (SELECT * FROM
             (SELECT * FROM BP
              WHERE BP.tm ≤ t AND BP.tm ≥ t − 180) AS T_1
           ORDER BY T_1.tm DESC FETCH FIRST 3 ROWS ONLY) AS T_2
         ORDER BY T_2.tm ASC FETCH FIRST 2 ROWS ONLY) AS T_3
       WHERE T_3.systolic ≥ 135

Q_F^E: SELECT * FROM
         (SELECT * FROM
           (SELECT * FROM Alert AL
           WHERE AL.tm ≥ t AND AL.tm ≤ t + 180) AS T_1
         WHERE T_1.tm ≥ t AND bp IN Q^B) AS T_2
       WHERE bp.systolic ≥ 135 AND T_2.tm ≥ t
```

Fig. 3. Provenance queries

Backward Provenance	Forward Provenance

SELECT * **FROM** *I* **WHERE** id **IN**
(**SELECT** *in.id* **FROM** *M*
WHERE *out.tm* = $\langle$*O.tm*$\rangle$)
	SELECT * **FROM** *O* **WHERE** id **IN**
	(**SELECT** *out.id* **FROM** *M*
	WHERE *in.tm* = $\langle$*I.tm*$\rangle$)

(a) Query templates for Alternative 1

	Backward Provenance	Forward Provenance

T	**SELECT** * **FROM** *I*
	WHERE *I.tm* $\leq$ ($\langle$*O.tm*$\rangle$ - $\langle t_b \rangle$)
	AND *I.tm* $\geq$ ($\langle$*O.tm*$\rangle$ - $\langle t_e \rangle$)
		SELECT * **FROM** *O* **WHERE**
		O.tm $\geq$ ($\langle$*I.tm*$\rangle$ + $\langle t_b \rangle$)
		AND *O.tm* $\leq$ ($\langle$*I.tm*$\rangle$ + $\langle t_e \rangle$)

S	**SELECT** * **FROM**
	(**SELECT** * **FROM** *I*
	WHERE *I.tm* $\leq$ $\langle$*O.tm*$\rangle$
	ORDER BY *I.tm* **DESC**
	FETCH FIRST $\langle s_e \rangle$ **ROWS ONLY**) **AS** *T*
	ORDER BY *T.tm* **ASC**
	FETCH FIRST ($\langle s_e \rangle - \langle s_b \rangle + 1$)
	ROWS ONLY
		SELECT * **FROM** *O* **WHERE**
		O.tm $\geq$ $\langle$*I.tm*$\rangle$
		AND $\langle$*I*$\rangle$ **IN** Q_S^B(O)

V	**SELECT** * **FROM** *I*
	WHERE *I.tm* $\leq$ $\langle$*O.tm*$\rangle$ **AND**
	$\langle$*I.attr*$\rangle$ $\geq$ $\langle v_b \rangle$ **AND** $\langle$*I.attr*$\rangle$ $\leq$ $\langle v_e \rangle$
		SELECT * **FROM** *O* **WHERE**
		$\langle$*I.attr*$\rangle$ $\geq$ $\langle v_b \rangle$ **AND**
		$\langle$*I.attr*$\rangle$ $\leq$ $\langle v_e \rangle$ **AND**
		O.tm $\geq$ $\langle$*I.tm*$\rangle$

(b) Query templates for Alternative 2

Fig. 4. Provenance query templates

retrieve the blood-pressure readings that resulted the alert. We call this a *backward* provenance query which given an output data item (e.g. medical alert) of some analysis, the query retrieves all the inputs (and intermediate results) contributing to its generation. Furthermore, John's doctor and the medical researchers studying John's condition also find *forward* provenance queries to be equally useful. In a forward provenance query, given a (possibly abnormal) blood-pressure reading, the query should return the alerts (if any) that might have been generated partially due to this reading.

Clearly, in healthcare (but also in domains like banking/finance) timely reaction to events (e.g., alerts) is paramount and provenance information plays a big role in deciding the type of this reaction. Therefore, the *efficiency* of provenance queries quickly becomes an issue. Currently, there are two alternative strategies used to address efficiency issues:

Alternative 1: An obvious alternative is to *materialize* a relation like *BP2Alert* (shown in Figure 2(c)) to encode the provenance relationship (expressed by the corresponding TVC rule) between readings and alerts. Then, efficient evaluation of backward/forward provenance queries essentially amounts to evaluating simple SQL queries like Q_B^M and Q_F^M (shown in Figure 3 with the generic template to generate such queries shown in Figure 4(a)) over the *BP2Alert* relation. Obviously, such an approach requires (a) to persist a relation like *BP2Alert*, and (b) to create and maintain a separate index for each of the attribute columns (e.g., B+-trees, hash indexes), to efficiently evaluate backward/forward provenance queries. In the literature, models for data provenance that use annotations (e.g., the initial version of Mondrian [11], DBNotes [17]), employ this alternative and rely essentially on materialization and general-purpose indexes to answer provenance queries.

Alternative 2: The second alternative avoids provenance data materialization and instead relies on *encoding* the provenance relationship between readings and alerts directly into the backward/forward provenance queries (see Q_B^E and Q_F^E in Figure 3) which are evaluated over the base *BP* and *Alert* relations. In more detail, Figure 4(b) shows the SQL provenance queries corresponding to each of the TVC primitives of the previous section, both for the case of forward and that of backward provenance. As an example, in the second row of Figure 4(b), the left query *implements* the TVC sequence rule of the previous section, while the right query *implements* the inverse rule (not shown) that given an input item it returns all the output items whose sequence rule includes this input item. In terms of the complex queries in Figure 3, these result in by composing the SQL queries of the corresponding primitives. So, query Q_B^E results in by composing the backward provenance queries for all three primitives, since all the primitives are involved in the corresponding "Prehypertension" alert rule. Although complex queries can result in easily, even while expressing simple provenance relationships between data items through the TVC provenance model, this complexity is not an artifact of the specific model. Indeed complex provenance queries are common in other provenance models found in the literature [11][13].

Both alternatives have their shortcomings. Alternative 1 imposes the seemingly innocent requirement to persist provenance data. Yet, this requirement has severe implications in terms of storage as recent studies have shown that provenance data are often orders of magnitude larger than base data [18]. Even worse, multiple types of inter-relationships between two data sets might exist, each requiring each own provenance data relation. Therefore, any solution that reduces, or eliminates, provenance data materialization is desirable. The alternative also requires that separate indexes are maintained to efficiently evaluate the different types of provenance queries. This latter requirement is a consequence of the inherent *directional* nature of general-purpose indexes: Given a medical alert, an index can retrieve all the blood pressure readings that are associated with the alert. However, starting from a reading we cannot use the same index to retrieve its generated alert. Such a functionality assumes that the index is somehow *bi-directional*. Using a single bi-directional index, *without compromising query performance*, has obvious advantages both in terms of maintenance costs and space overhead.

The shortcomings of Alternative 2 are equally profound. In real life applications, data inter-relationships are inherently complex. In turn, the provenance queries encoding these inter-relationships are quite involved in nature. Provenance queries (often with deep nesting) are usually hard to optimize, even by the most sophisticated optimizers. Therefore, the queries often suffer from poor performance. Customized indexes, like the one introduced in MMS [13] or later in Mondrian [14], can improve the efficiency of the queries. However, their use is often limited to particular provenance models.

III. A GENERIC PROVENANCE INDEX STRUCTURE

To address the shortcomings of the existing alternatives, we argue for a *generic* index structure for the efficient evaluation of provenance queries. To our knowledge this is the *first* work to address the *open problem* of efficient provenance query evaluation. In what follows, we highlight some of the key (desirable) characteristics of the proposed index structure.

Model Independence: Ultimately, the generality of the index is determined by its ability to be used in a variety of provenance models. Since provenance can abstractly be modeled as a binary relation between data items and processing nodes of an analytics workflow (in the case of workflow provenance), and between different data items (in the case of data provenance), we argue here for a structure that can be used to index arbitrary binary relations. As such, our index is independent of both the specific of the provenance model and the complexity of its corresponding provenance queries.

Index Duality: As further evidence of its generality, it would be desirable if the same index can be used to evaluate provenance queries of different types. That is, we argue here for a *single* index structure that exhibits a duality property in that it can be used to evaluate both forward and backward provenance queries. Notice that no existing general-purpose index (e.g., B+-trees, hash index) or provenance-aware index (e.g., Q-index [13]) exhibits this property.

Index performance: Finally, the performance of the index should be such that it outperforms conventional indexes, like B+-trees, both in terms of time and space. As such, we argue for an index that relies on state-of-the-art data structures that (a) have provable performance (b) can be compressed to reduce the memory/disk requirements; and (c) can take advantage of the latest trends in hardware (e.g., multi-core CPUs) to improve efficiency.

An index with these characteristics is currently under development and part of the provenance infrastructure we are building for the IBM Century project and beyond. Yet, due to its generality, we note that our index is *directly* applicable and can be used in other existing provenance-enabled systems like Trio [2], GridDB [19] and Zoom [20].

REFERENCES

[1] P. Buneman and W.-C. Tan, "Provenance in databases," in *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2007, pp. 1171–1173.

[2] J. Widom, "Trio: A system for integrated management of data, accuracy, and lineage," in *CIDR*, 2005, pp. 262–276.

[3] W. C. Tan, "Provenance in databases: Past, current, and future," *IEEE Data Eng. Bull.*, vol. 30, no. 4, pp. 3–12, 2007.

[4] R. Bose and J. Frew, "Lineage retrieval for scientific data processing: a survey," *ACM Comput. Surv.*, vol. 37, no. 1, pp. 1–28, 2005.

[5] M. Blount et. al., "Century: Automated aspects of patient care," in *Proceedings of the 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, 2007, pp. 504–509.

[6] M. Jayapandian, A. Chapman, V. G. Tarcea, C. Yu, A. Elkiss, A. Ianni, B. Liu, A. Nandi, C. Santos, P. Andrews, B. Athey, D. J. States, and H. V. Jagadish, "Michigan molecular interactions (mimi): putting the jigsaw puzzle together," *Nucleic Acids Research*, vol. 35, no. Database-Issue, pp. 566–571, 2007.

[7] A. Misra, M. Blount, A. Kementsietsidis, D. Sow, and M. Wang, "Advances and challenges for scalable data provenance in stream processing systems," in *IPAW*, 2008.

[8] Y. L. Simmhan, B. Plale, D. Gannon, and S. Marru, "Performance evaluation of the karma provenance framework for scientific workflows," in *IPAW*, 2006, pp. 222–236.

[9] S. Bowers, T. M. McPhillips, and B. Ludäscher, "Provenance in collection-oriented scientific workflows," *Concurr. Comput. : Pract. Exper.*, vol. 20, no. 5, pp. 519–529, 2008.

[10] P. Buneman, S. Khanna, and W. C. Tan, "Why and where: A characterization of data provenance," in *ICDT*, 2001, pp. 316–330.

[11] F. Geerts, A. Kementsietsidis, and D. Milano, "Mondrian: Annotating and querying databases through colors and blocks," in *ICDE*, 2006, p. 82.

[12] T. J. Green, G. Karvounarakis, and V. Tannen, "Provenance semirings," in *PODS*, 2007, pp. 31–40.

[13] D. Srivastava and Y. Velegrakis, "Intensional associations between data and metadata," in *SIGMOD Conference*, 2007, pp. 401–412.

[14] M. Mavromatis, "Indexing in the mondrian annotation management system," School of Informatics, University of Edinburgh, Tech. Rep. EDI-INF-IM060399, 2006.

[15] S. B. Davidson, "On provenance and user views in scientific workflows," in *DBIR2008 (Keynote speech)*, 2008.

[16] M. Wang, M. Blount, J. Davis, A. Misra, and D. M. Sow, "A time-and-value centric provenance model and architecture for medical event streams," in *HealthNet*, 2007, pp. 95–100.

[17] D. Bhagwat, L. Chiticariu, W. C. Tan, and G. Vijayvargiya, "An annotation management system for relational databases," in *VLDB*, 2004, pp. 900–911.

[18] A. P. Chapman, H. Jagadish, and P. Ramanan, "Efficient provenance storage," in *SIGMOD Conference*, 2008.

[19] D. T. Liu and M. J. Franklin, "The design of griddb: A data-centric overlay for the scientific grid," in *VLDB*, 2004, pp. 600–611.

[20] O. Biton, S. C. Boulakia, and S. B. Davidson, "Zoom*userviews: Querying relevant provenance in workflow systems," in *VLDB*, 2007, pp. 1366–1369.